Mobile Testing

Типы мобильных приложений

Типы мобильных приложений делятся на 3 группы по особенностям работы: ***нативные, гибридные, веб-приложения.***

Нативные

Создаются для конкретной операционной системы (iOS, Android, Windows). Для охвата аудитории нужно разработать несколько отдельных приложений для разных операционных систем. Они могут выполнять одни функции, иметь одинаковый дизайн, но будут разными программами. Эта необходимость увеличивает срок работы над проектом и бюджет разработки.

Нативные сервисы могут работать независимо от подключения к интернету, хотя часть из них требует наличия подключения. Они занимают меньше памяти, работают быстро, тратят меньше заряда батареи. Могут получить доступ к аппаратной части телефона по разрешению владельца.

Достоинства:

- Превосходная производительность: нативные приложения выполняются плавно и без зависаний даже в случаях повышенной нагрузки на графический процессор и интеграции сложных вычислений.

- Наличие доступа к индивидуальным возможностям платформы: самое лучшее в этом типе приложений — это то, что они обеспечивают доступ к встроенным возможностям устройств или конкретной платформы.
- Нативный пользовательский интерфейс: плавный опыт использования обеспечивается благодаря тому, что приложения создаются в соответствии со стандартами платформы.

Недостатки:

- Необходимы две команды разработчиков: нативные приложения для Android обычно создаются на Java или Kotlin, в то время как приложения для iOS разрабатываются на Objective-C или Swift, в связи с чем вам потребуется нанять команду, имеющую опыт работы именно с этими языками.
- Высокая стоимость разработки: нативные приложения идеально подходят для крупных корпораций с большими бюджетами. Поскольку вам нужно разрабатывать каждое приложение с нуля, то для его итоговой экономической успешности потребуется много ресурсов, в том числе времени.

Гибридные

Занимают промежуточную позицию между нативными и веб программами. Имеют ограниченный доступ к аппаратной части смартфона (камера, микрофон, геолокация, адресная книга). Требуют подключения к интернету, поскольку загружают контент из внешнего источника, размещенного на сервере. Большинство промо-сервисов относится к этой категории.

У гибридных программ есть недостатки. Эти сервисы оперируют малым объемом информации. Дизайн не адаптируется к размеру и расширению экрана, что может вызвать неудобства.

Достоинства:

- Быстрый вывод на рынок: гибридные приложения разрабатываются быстрее, так как в ходе процесса используются стандартные веб-технологии, которые легко обслуживать в долгосрочной перспективе.
- Доступ к возможностям устройств: используя гибридные приложения, вы также можете использовать нативные возможности целевых устройств.

- Дистрибуция для нескольких платформ: этот вид приложений распространяется через оба магазина, что позволяет охватить большее число пользователей.

Недостатки:

- Производительность: их производительность ниже, чем у нативных вариантов, так как зависит от качества процессов, отображающих UI и выполняющих код. Поэтому чем быстрее использующее приложение устройство, тем выше его производительность.
- Интеграция сторонних сервисов: Вы не можете разрабатывать гибридное приложение на одном только JavaScript. Вам потребуется интегрировать такие фреймворки для гибридной разработки, как Cordove, Ionic или React Native, каждый из которых требует определенных усилий для освоения.

Веб - приложения

Веб приложения являются адаптацией сайтов для пользователей смартфонов. Они создаются, чтобы посетители могли заходить на сайт в любое время, даже без доступа к персональному компьютеру или ноутбуку. Некоторые веб сервисы необходимо скачивать и устанавливать. Другие запускаются автоматически при заходе на сайт через мобильный браузер.

Достоинства:

- Совместимость с несколькими платформами: создав web-приложение вы можете тут же запускать его на любой платформе, не вкладывая дополнительных средств и времени в дополнительную разработку.
- Мгновенные обновления: при использовании гибридных приложений пользователи всегда имеют доступ к их последним версиям и скачивать обновления им не приходится.
- Использование распространенных технологий: поскольку web-приложения можно разрабатывать при помощи различных технологий, выбор наиболее подходящей компании-разработчика не представляет для стартапов сложности.

Недостатки:

- Ограниченный доступ к нативным функциям платформы: web-приложения не имеют доступа к встроенным возможностям устройств, таким как

камера, хранилище, контакты и прочее.

- Базовая производительность: эти приложения работают плавно в простых случаях применения, таких как новые издательства и онлайн магазины.

Требования к разработке приложений на различных платформах

Android

Material design - это всеобъемлющее руководство по визуальному дизайну, дизайну движения и взаимодействия на разных платформах и устройствах.

Android предоставляет следующие функции, которые помогут вам создавать приложения material design:

- Тема приложения material Design для оформления всех ваших виджетов пользовательского интерфейса
- Виджеты для сложных представлений, таких как списки и карточки
- Новые API для пользовательских теней и анимации

Требования к разработке приложений

Тема материалов и виджеты

Чтобы воспользоваться преимуществами функций material, таких как оформление стандартных виджетов пользовательского интерфейса, и упростить определение стиля вашего приложения, примените к своему приложению тему на основе материалов.

Тема темного материала

Тема Light material

Для получения дополнительной информации см. раздел Как применить тему материала.

Чтобы предоставить пользователям привычный интерфейс, используйте наиболее распространенные шаблоны пользовательского интерфейса material:

- Продвигайте основное действие вашего пользовательского интерфейса с помощью плавающей кнопки действия (FAB).

- Покажите свой бренд, навигацию, поиск и другие действия на панели приложений.
- Показывайте и скрывайте навигацию вашего приложения с помощью панели навигации.
- Используйте один из многих других компонентов material для компоновки и навигации вашего приложения, таких как сворачивающиеся панели инструментов, вкладки, нижняя панель навигации и многое другое. Чтобы увидеть их все, ознакомьтесь с каталогом Material Components для Android

И по возможности используйте предопределенные значки материалов. Например, кнопка навигации "меню" для вашего навигационного ящика должна использовать стандартный значок "гамбургер". Список доступных значков см. в разделе Значки Material Design. Вы также можете импортировать значки SVG из библиотеки значков материалов с помощью Vector Asset Studio от Android Studio.

Тени и карты высот

В дополнение к свойствам X и Y, представления в Android имеют свойство Z. Это новое свойство представляет высоту вида, которая определяет:

- Размер тени: виды с более высокими значениями Z отбрасывают большие тени.
- Порядок рисования: виды с более высокими значениями Z отображаются поверх других видов.

Возвышение часто применяется, когда ваш макет включает макет на основе карточек, который помогает отображать важные фрагменты информации внутри карточек, которые обеспечивают материальный вид. Вы можете использовать `CardView` виджет для создания карточек с высотой по умолчанию. Дополнительные сведения см. в разделе Создание макета на основе карточек.

Дополнительные сведения о добавлении высоты к другим видам см. в разделе Создание теней и видов обрезки.

Анимация

Новые API анимации позволяют создавать пользовательские анимации для сенсорной обратной связи в элементах управления пользовательского интерфейса, изменения состояния просмотра и переходов действий.

Эти API позволяют:

- Реагируйте на события **касания в своих представлениях с помощью анимации обратной** связи.
- Скрывать и **показывать виды с помощью круговой** анимации отображения.
- Переключайтесь между **действиями с помощью пользовательских анимаций перехода**.
- Анимируйте изменения в одном или нескольких свойствах представления с помощью анимации изменения состояния представления.
- Создавайте более естественные анимации с **изогнутым движением**.
- Показывать анимацию в чертежах списка состояний между изменениями состояния просмотра.

Анимация сенсорной обратной связи встроена в несколько стандартных представлений, таких как кнопки. Новые API позволяют настраивать эти анимации и добавлять их в свои пользовательские представления.

Для получения дополнительной информации см. Обзор анимации.

Чертежи

Эти новые возможности для рисования помогут вам создавать приложения для material design:

- **Векторные рисунки** масштабируются без потери четкости и идеально подходят для одноцветных иконок в приложении. Узнайте больше о векторных чертежах.
- **Рисованное тонирование** позволяет определять растровые изображения как альфа-маску и окрашивать их цветом во время выполнения. Посмотрите, как добавить оттенок к чертежам.
- **Извлечение цвета** позволяет автоматически извлекать выделяющиеся цвета из растрового изображения. Посмотрите, как выбирать цвета с помощью API палитры.

iOS

Рекомендации, которые помогут создать отличный интерфейс для любой платформы Apple.

Кнопки

Универсальные и настраиваемые кнопки предоставляют пользователям простые и знакомые способы выполнения задач в вашем приложении. В общем, кнопка объединяет три атрибута, чтобы четко передать свою функцию:

- **Стиль.** Визуальный стиль, основанный на размере, цвете и форме.
- **Содержание.** Символ (или значок интерфейса), текстовая метка или и то, и другое, которые отображаются на кнопке, чтобы передать ее назначение.
- **Роль.** Определяемая системой роль, которая определяет семантическое значение кнопки и может повлиять на ее внешний вид.

Лучшие практики

Когда кнопки мгновенно узнаваемы и понятны, приложение, как правило, интуитивно понятное и хорошо спроектированное.

Сделайте кнопки удобными для выбора людьми.

Убедитесь, что каждая кнопка четко сообщает о своем назначении.

Содержание

Создавайте содержимое кнопок, чтобы люди сразу понимали, что делает кнопка. Например, кнопка, которая позволяет пользователям добавлять товары в корзину, может использовать надпись “Добавить в корзину”.

Добавляйте дополнительный текст под надписью только в том случае, если он содержит полезные сведения.

Настройте кнопку для отображения индикатора активности, когда вам нужно предоставить отзыв о действии, которое не выполняется мгновенно.

Роль

Системная кнопка может выполнять одну из следующих ролей:

- **Нормальный.** Никакого конкретного значения.
- **Основное.** Кнопка — это кнопка по умолчанию - кнопка, которую люди, скорее всего, выберут.

- **Отмена.** Кнопка отменяет текущее действие.
- **Разрушительный.** Кнопка выполняет действие, которое может привести к уничтожению данных.

Назначьте основную роль кнопке, которую люди, скорее всего, выберут. Когда основная кнопка реагирует на клавишу возврата, пользователям становится проще быстро подтвердить свой выбор. Кроме того, когда кнопка находится во временном представлении — например, на листе, в редактируемом представлении или в предупреждении, — назначение ей основной роли означает, что представление может автоматически закрываться, когда пользователи нажимают Return.

Не назначайте основную роль кнопке, которая выполняет деструктивное действие, даже если это действие является наиболее вероятным выбором. Из-за его визуальной значимости люди иногда выбирают основную кнопку, не прочитав ее сначала. Помогите людям избежать потери содержимого, назначив основную роль неразрушающим кнопкам.

Жесты с сенсорным экраном

Жесты - это ключевой способ взаимодействия людей со своими устройствами с сенсорными экранами, позволяющий установить тесную личную связь с контентом и улучшить ощущение непосредственного управления объектами на экране.

Помимо использования сенсорного экрана, пользователи также могут жестикулировать с помощью таких устройств, как трекпад, мышь, пульт дистанционного управления или игровой контроллер. Например, пользователи могут использовать трекпад для взаимодействия со своим iPad или Mac, а также использовать игровой контроллер для взаимодействия с iPhone, iPad, Mac и Apple TV. Инструкции по вводу данных с этих устройств см. в разделе Указывающие устройства, пульты дистанционного управления и игровые контроллеры.

Все устройства с сенсорным экраном используют базовые жесты, такие как нажатие, свайп и перетаскивание. Некоторые платформы определяют дополнительные жесты; например, iOS и iPadOS поддерживают сжатие и поворот. При внедрении жестов с сенсорным экраном в свой интерфейс необходимо понимать поведение стандартных жестов каждой платформы, чтобы обеспечить привычный и согласованный интерфейс.

Лучшие практики

В общем, реагируйте на жесты так, как это делают другие приложения. Люди ожидают, что большинство жестов будут работать одинаково независимо от их текущего контекста.

Задавайте пользовательские жесты только при необходимости. Если вы решите задать пользовательский жест, убедитесь, что он:

- Легко находить и выполнять
- Не слишком похоже на жесты, которые люди уже знают
- Не единственный способ выполнить важное действие в вашем приложении

Убедитесь, что жесты применяются к соответствующему контенту.В целом, жесты должны применяться к контенту, с которым в данный момент взаимодействуют пользователи, например, к выбранному элементу, активному виду в окне или области поверх элемента, например фотографии.

Обрабатывайте жесты как можно быстрее.

Включите жесты быстрого доступа, чтобы дополнять стандартные жесты, а не заменять их. Людям нужны простые, знакомые способы навигации и выполнения действий, даже если это означает одно или два дополнительных нажатия. Например, в приложении, которое поддерживает навигацию по иерархии экранов, пользователи ожидают найти кнопку "Назад" на панели навигации, которая позволяет им вернуться к предыдущему экрану одним нажатием.

Избегайте вмешательства в общесистемные жесты по краям экрана. Для получения рекомендаций для разработчиков см. Свойство preferredScreenEdgesDeferringSystemGestures UIViewController.

Меню

Система предоставляет несколько типов меню, которые поддерживают различные варианты использования, такие как:

- Кнопка, подобная всплывающей кнопке или выпадающей кнопке, которая открывает меню опций, непосредственно связанных с ее действием
- Скрытое контекстное меню, которое пользователи могут открыть для доступа к небольшому количеству часто используемых действий, относящихся к их текущему представлению или задаче

- Меню в строке меню приложения macOS, содержащее все команды, которые пользователи могут выполнять в приложении

Независимо от типа, во всех меню перечислены один или несколько *пунктов меню*, каждый из которых представляет команду, параметр или состояние, влияющие на текущий выбор или контекст. Когда пользователи выбирают пункт меню, происходит действие, и меню обычно закрывается.

Панели инструментов

Панель инструментов обеспечивает удобный доступ к часто используемым командам и элементам управления, которые выполняют действия, относящиеся к текущему представлению.

В зависимости от платформы панель инструментов может выглядеть и работать по-разному.

- В iOS панель инструментов отображается в нижней части экрана. Панели инструментов iOS не настраиваются и не поддерживают группировку.
- В iPadOS и macOS панель инструментов отображается в верхней части экрана или окна. Обе платформы поддерживают настраиваемые панели инструментов и сгруппированные элементы панели инструментов. В macOS пользователи могут скрывать панель инструментов приложения.
- В watchOS сама панель инструментов не отображается, но кнопка панели инструментов может отображаться в верхней части прокрутки. Обычно кнопка панели инструментов остается скрытой за панелью навигации, пока пользователи не откроют ее, прокрутив вверх.

Лучшие практики

Предоставьте элементы панели инструментов, которые поддерживают основные задачи, выполняемые людьми.

Избегайте отображения слишком большого количества элементов панели инструментов. Люди должны уметь различать и активировать каждый элемент, поэтому вы не хотите перегружать панель инструментов.

Рассмотрите возможность группировки элементов панели инструментов там, где это поддерживается.

Убедитесь, что значение каждого элемента панели инструментов понятно. Людям не нужно гадать или экспериментировать, чтобы выяснить, что делает

элемент. Для каждого элемента укажите простой, узнаваемый символ или значок интерфейса или короткую описательную метку.

Предпочитайте системные символы или значки интерфейса. Системные символы знакомы, автоматически получают соответствующую окраску и постоянно реагируют на взаимодействие с пользователем и динамичность.

Предпочитайте одинаковый внешний вид для всех элементов панели инструментов. Панели инструментов выглядят лучше и их легче понять, когда все элементы имеют одинаковый визуальный стиль.

Если элемент панели инструментов переключается между двумя состояниями, убедитесь, что элемент четко передает текущее состояние.

В приложениях для iPadOS и macOS рассмотрите возможность предоставления пользователям возможности настраивать панель инструментов.

Будьте готовы к прозрачности панели инструментов, когда под ней появляется содержимое.

Windows Phone

Рекомендации по разработке фреймворка

В этом разделе приведены рекомендации по разработке библиотек, которые расширяют .NET Framework и взаимодействуют с ним. Цель состоит в том, чтобы помочь разработчикам библиотек обеспечить согласованность API и простоту использования путем предоставления унифицированной модели программирования, которая не зависит от языка программирования, используемого для разработки. Мы рекомендуем следовать этим рекомендациям по проектированию при разработке классов и компонентов, расширяющих .NET Framework. Непоследовательный дизайн библиотеки отрицательно влияет на производительность разработчиков и препятствует внедрению.

Рекомендации организованы в виде простых рекомендаций, перед которыми стоят термины `Do`, `Consider`, `Avoid`, и `Do not`. Эти рекомендации призваны помочь разработчикам библиотек классов понять компромиссы между различными решениями. Могут быть ситуации, когда для правильного проектирования библиотеки требуется, чтобы вы нарушали эти рекомендации

по проектированию. Такие случаи должны быть редкими, и важно, чтобы у вас была четкая и убедительная причина для вашего решения.

Рекомендации

Рекомендации по именованию Содержит рекомендации по присвоению имен сборкам, пространствам имен, типам и элементам в библиотеках классов.

Рекомендации по разработке типов Содержит рекомендации по использованию статических и абстрактных классов, интерфейсов, перечислений, структур и других типов.

Рекомендации по разработке участников Содержит рекомендации по разработке и использованию свойств, методов, конструкторов, полей, событий, операторов и параметров.

Проектирование с учетом расширяемости Рассматриваются механизмы расширения, такие как создание подклассов, использование событий, виртуальных членов и обратных вызовов, а также объясняется, как выбрать механизмы, которые наилучшим образом соответствуют требованиям вашей фреймворка.

Рекомендации по разработке исключений Описывает рекомендации по проектированию для разработки, создания и перехвата исключений.

Рекомендации по использованию Описывает рекомендации по использованию общих типов, таких как массивы, атрибуты и коллекции, поддержке сериализации и перегрузке операторов равенства.

Общие шаблоны проектирования Содержит рекомендации по выбору и реализации свойств зависимостей.

Нефункциональное тестирование мобильных приложений

Конфигурационное тестирование

Тестирование совместимости имеет самый высокий стек, когда дело доходит до тестирования мобильных приложений. Цель теста на совместимость мобильного приложения, как правило, состоит в том, чтобы ключевые функции приложения работали должным образом на конкретном устройстве. Сама

совместимость должна занимать всего несколько минут и может быть спланирована заранее.

Это не будет легкой задачей, решить, какие тесты на совместимость мобильных устройств следует выполнить (поскольку тестирование со всеми доступными устройствами просто невозможно). Поэтому подготовьте тестовую матрицу с каждой возможной комбинацией и расставьте приоритеты для клиента.

Пример тестовых сценариев –

- Убедитесь, что поиск авиабилетов успешно выполняется на устройстве Android.
- Убедитесь, что поиск авиабилетов успешно выполнен для Apple iPad.

Тестирование совместимости с функциями телефона

Тестирование направленное на определение возможности приложения использовать функции телефона

- Приложение поддерживает функции телефона - управление жестами, голосовое управление и т.д.
- Приложение работает в сети 5 g
- Существует возможность загружать файлы в приложение из памяти телефона и сохранять файлы в память телефона

Тестирование локализации

В настоящее время большинство приложений предназначено для глобального использования, и очень важно заботиться о региональных следах, таких как языки, часовые пояса и т. Д. Важно проверить функциональность приложения, когда кто-то меняет часовой пояс. Необходимо учитывать, что иногда западные дизайны могут не работать с аудиторией из восточных стран или наоборот.

Пример тестовых сценариев –

- Убедитесь в отсутствии проблем с пользовательским интерфейсом или усечением данных, когда мы используем мобильное приложение на разных языках (или, скажем, не на английском языке).

- Убедитесь, что изменения часового пояса корректно обрабатываются для вашего мобильного приложения.

Лабораторные испытания

Лабораторные испытания, обычно проводимые сетевыми операторами, выполняются путем моделирования всей беспроводной сети. Этот тест выполняется для обнаружения каких-либо сбоев, когда мобильное приложение использует передачу голоса и / или данных для выполнения некоторых функций.

Пример тестовых сценариев –

- Убедитесь, что нет никаких сбоев, когда у клиента есть голосовой чат со службой поддержки.

Тестирование производительности

Мобильный тест производительности охватывает производительность клиентских приложений, производительность сервера и производительность сети. Важно убедиться, что сценарии тестирования производительности охватывают все эти области. С помощью инструментов тестирования производительности нетрудно идентифицировать существующие сети, серверы и узкие места серверных приложений, учитывая предопределенную нагрузку и сочетание транзакций.

Пример тестовых сценариев –

- Убедитесь, что проверка доступного рейса занимает только разумное количество времени.
- Убедитесь, что во время проверки доступности рейса мобильный телефон работает нормально и не зависает.

Убедитесь, что проверка доступного рейса занимает только разумное количество времени.

Убедитесь, что во время проверки доступности рейса мобильный телефон работает нормально и не зависает.

Стресс-тестирование

Стресс-тестирование является обязательным условием для обнаружения исключений, зависаний и взаимоблокировок, которые могут остаться незамеченными во время функционального тестирования и тестирования пользовательского интерфейса. Вот список некоторых критериев для стресс-тестирования –

- Загрузите в свое приложение как можно больше данных, чтобы попытаться достичь его предела.
- Выполняйте одни и те же операции снова и снова.
- Выполняйте повторные операции на разных скоростях – очень быстро или очень медленно.
- Оставьте приложение работающим в течение длительного периода времени, одновременно взаимодействуя с устройством и просто оставляя его бездействующим, или выполняя некоторую автоматическую задачу, которая занимает много времени, например, слайд-шоу.
- Случайно отправлять экранные нажатия и нажатия клавиш в вашем приложении.
- На вашем устройстве должно быть запущено несколько приложений, чтобы вы могли часто переключаться между приложением и другими приложениями на устройстве.

Загрузите в свое приложение как можно больше данных, чтобы попытаться достичь его предела.

Выполняйте одни и те же операции снова и снова.

Выполняйте повторные операции на разных скоростях – очень быстро или очень медленно.

Оставьте приложение работающим в течение длительного периода времени, одновременно взаимодействуя с устройством и просто оставляя его бездействующим, или выполняя некоторую автоматическую задачу, которая занимает много времени, например, слайд-шоу.

Случайно отправлять экранные нажатия и нажатия клавиш в вашем приложении.

На вашем устройстве должно быть запущено несколько приложений, чтобы вы могли часто переключаться между приложением и другими приложениями на устройстве.

Пример тестовых сценариев –

- Проверьте, что 1000 пользователей получают доступ к мобильному приложению для поиска внутренних рейсов.
- Проверьте, что 1000 пользователей получают доступ к мобильному приложению для поиска международных рейсов.

Тестирование безопасности/ конфиденциальности

Уязвимости в отношении политик взлома, аутентификации и авторизации, безопасности данных, управления сеансами и других стандартов безопасности должны быть проверены как часть тестирования безопасности мобильных приложений. Приложения должны шифровать имя пользователя и пароли при аутентификации пользователя по сети.

Один из способов тестирования сценариев, связанных с безопасностью, заключается в маршрутизации данных вашего мобильного устройства через прокси-сервер, такой как OWASP Zed Attack Proxy, и поиске уязвимостей.

Пример тестовых сценариев –

- Убедитесь, что приложение не работает с одинаковыми учетными данными на двух разных мобильных устройствах.
- Убедитесь, что сеанс автоматически истекает, если он остается неактивным в течение более 15 минут.

Тестирование утечки памяти

Мобильные устройства имеют очень ограниченную память по сравнению с другими компьютерами, и мобильные операционные системы имеют поведение по умолчанию для завершения приложений, которые используют чрезмерную память и вызывают плохое взаимодействие с пользователем.

Тестирование памяти исключительно важно для мобильных приложений, чтобы гарантировать, что каждое приложение поддерживает оптимизированное использование памяти на протяжении всего пути пользователя. Рекомендуется проводить тестирование памяти на реальном целевом устройстве, поскольку архитектура системы отличается от эмулятора до реального устройства.

Пример тестовых сценариев –

- Сделайте проверку доступности рейса в десять раз и запишите увеличение использования памяти для каждой проверки.
- Продолжайте работу приложения в течение десяти минут и наблюдайте, остается ли использование памяти стабильным.

Тестирование энергопотребления

Существует несколько типов батарей, используемых в различных мобильных устройствах (например, никель-кадмиевый / литий-ионный / никель-металлический гибрид). Хотя мы сосредоточены на тестировании энергопотребления, мы обязаны измерять состояние батареи на каждом уровне активности. Это даст нам лучшее понимание энергопотребления для отдельного приложения.

Тест энергопотребления может быть выполнен вручную; Также на рынке есть несколько бесплатных инструментов, таких как Trepn Profiler, Power Tutor и Nokia Energy Profiler. Это приложения, которые могут отображать энергопотребление в реальном времени на смартфоне или планшете.

Пример тестовых сценариев –

- Используйте мобильное приложение для поиска доступности рейса и убедитесь, что энергопотребление остается минимальным.
- Держите мобильное приложение в идеальном состоянии; убедитесь, что нет энергопотребления, когда нет активности для приложения.

Тестирование прерываний

Приложение во время работы может столкнуться с несколькими перебоями, такими как входящие вызовы или отключение и восстановление покрытия сети. Это снова можно отличить для –

- Входящие и исходящие SMS и MMS
- Входящие и исходящие звонки
- Входящие уведомления
- Удаление батареи
- Вставка и удаление кабеля для передачи данных

Пример тестовых сценариев –

- Убедитесь, что проверка доступности рейса приостановлена и возобновлена после получения входящего вызова.
- Убедитесь, что пользователь может отклонить вызов во время использования приложения, а затем снова возобновить работу того же приложения.

Юзабилити-тестирование

Юзабилити-тестирование оценивает приложение на основе следующих трех критериев для целевой аудитории:

- **Эффективность** . Точность и полнота, с которой указанные пользователи могут достичь определенных целей в конкретной среде.
- **Эффективность** – ресурсы, затраченные на точность и полноту достигнутых целей.
- **Удовлетворенность** – комфорт и приемлемость рабочей системы для ее пользователей и других людей, пострадавших от ее использования.

Эффективность . Точность и полнота, с которой указанные пользователи могут достичь определенных целей в конкретной среде.

Эффективность – ресурсы, затраченные на точность и полноту достигнутых целей.

Удовлетворенность – комфорт и приемлемость рабочей системы для ее пользователей и других людей, пострадавших от ее использования.

Очень важно провести юзабилити-тестирование на месте с самого раннего этапа разработки приложения, и оно не должно проводиться только после завершения приложения. Юзабилити-тестирование требует активного участия пользователей, и результаты могут повлиять на дизайн приложения, что очень трудно изменить на более поздних этапах проекта.

Пример тестовых сценариев –

- Проверка доступности рейса должна быть на главной странице.
- Рекламная реклама не должна отображаться в середине контента.

Тестирование установки

Тестирование установки подтверждает, что процесс установки проходит без каких-либо проблем.

Пример тестовых сценариев –

- Убедитесь, что процесс установки проходит гладко и не займет много времени.
- Убедитесь, что установка прошла успешно через корпоративный магазин приложений.

Тестирование удаления

Основы тестирования деинсталляции могут быть сведены в одну строку: «Деинсталляция должна охватывать данные, связанные с приложением, всего за один раз».

Пример тестовых сценариев –

- Убедитесь, что все файлы, связанные с приложением, успешно удалены после удаления.
- Если это приложение, в котором хранятся мультимедийные файлы (например, Whatsapp или Facebook), сохраните файлы даже после удаления приложения.

Тестирование обновлений

Мы должны быть очень осторожны с обновлениями мобильных приложений. Люди часто жалуются на то, что приложения не работают удовлетворительно после обновления. Поэтому очень важно, чтобы при тестировании обновлений мы определяли, что приложение будет работать так, как оно работало ранее. В двух словах, это ничего не должно сломать. Обновления мобильных приложений могут осуществляться двумя способами – **автоматическим обновлением** и **ручным обновлением** .

Пример тестовых сценариев –

- Убедитесь, что приложение успешно работает после автоматического обновления.
- Убедитесь, что процесс обновления отображается правильно.

Сертификационное тестирование

Чтобы получить сертификат соответствия, каждое мобильное устройство должно быть проверено на соответствие рекомендациям, установленным различными мобильными платформами.

Пример тестовых сценариев –

Убедитесь, что приложение придерживается политики телефонов iOS при установке на iPhone.

Убедитесь, что приложение придерживается политики телефонов Android при установке на Android.

Тестирование на отказ и восстановление

Исследование системы на возможность восстановления после возможных ошибок и сбоев.

Пример тестовых сценариев –

Убедитесь что приложение запускается после краша

Интеграционное тестирование

Это тип тестирования, при котором программные модули объединяются логически и тестируются как группа. Как правило, программный продукт состоит из нескольких программных модулей, написанных разными программистами. Целью такого тестирования является выявление проблем при взаимодействии между этими программными модулями и в первую очередь направлен на проверку обмена данными между этими самими модулями.

Пример тестовых сценариев –

Тестирование оплаты покупок в приложении

Тестирование авторизации через номер телефона/смс/google аккаунт

Тестирование на зависимость от сетей и каналов связи

Этот тип тестирования предназначен для проверки работоспособности приложения через различные сети и каналы связи

Пример тестовых сценариев –

Работа приложения через сети 3g, 4g, 5g, wi-fi